

Evaluation of Safe Control Policies for Mobile Robots in Dynamic Environments

Varun Murali, Niharika Arora, Ian Buckley

Abstract—With the growth of domestic robot industry, it is important to understand the navigation of mobile robots in dynamic environments. Use of mobile robots in the home can greatly improve the quality of life and is a growing field of research. With an increased interaction between humans and robots, and an increasingly shared workspace, it is important to study the safety of the robotic system and the extent of guarantees that can be made to minimize risk of injury. We present a nonlinear control approach to the navigation problem that achieves the objective of reaching pre-defined locations with provable guarantees of safety. Verification of the controller is performed first in MATLAB simulation, which shows that the controller is capable of driving the system to a pre-defined location while maintaining all required safety rules defined in it's environment. The system is implemented on a Segway RMP 200 based robot, demonstrating that the nonlinear navigation controller successfully traverses the gap between theory and practice.

I. INTRODUCTION

Safety is an important concern for mobile robots working in collaborative environments. Typical robotic systems separately deal with safety and task completion in a hierarchical fashion—a high level controller decides which control law to use based on the situation. Navigation is of fundamental concern for mobile robots. The ability to successfully avoid obstacles, both static and dynamic, while moving towards a goal location largely determines the utility of a mobile robotic platform. It is critical that robots avoid obstacles for their own safety as well as the safety of the people around them. We use the navigation problem to motivate the direct addition of safety into the controller for the robot.

After formulating the navigation control strategy, MATLAB simulation is used to verify that the navigation control strategy correctly drives the state of the robot to a target. After demonstrating that the robot successfully navigates to a target location, a trajectory is generated for a map using a probabilistic roadmap, and the navigation control strategy is used to drive the robot along the trajectory. After thoroughly substantiating the efficacy of the navigation control strategy in simulation, the controller is implemented in the ROS framework and evaluated on a Segway RMP 200 based mobile robot, verifying the navigation control strategy in a real world scenario.

II. APPROACH

Control Lyapunov functions and control barrier functions are formulated to individually accomplish the objectives of navigation and obstacle avoidance. The control Lyapunov and barrier functions are soft and hard constrained in a Quadratic Programming based controller to achieve safe navigation.

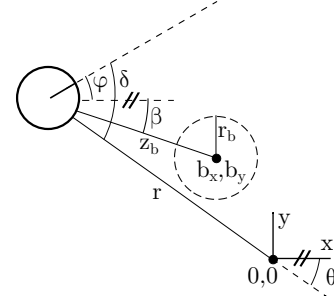


Fig. 1: The egocentric co-ordinate system of the robot is shown above. r is the radial distance of the robot from the origin, ϕ is the heading of the robot in the target frame, θ is the orientation of the target frame in the robot frame, and δ is the egocentric heading. β is the angle between the global x -axis and the vector joining the center of the robot to the obstacle.

Figure 1 shows the egocentric coordinate system of the robot and describes the variables used in the polar form of the unicycle dynamics modelling the robot, given by:

$$\begin{pmatrix} \dot{r} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} -v \cos(\delta) \\ \frac{v}{r} \sin(\delta) \end{pmatrix}. \quad (1)$$

$$\dot{\delta} = \frac{v}{r} \sin(\delta) + \omega \quad (2)$$

A. Control Lyapunov Functions

To design the control law, the standard quadratic Lyapunov function candidate is first considered:

$$V = \frac{1}{2}(r^2 + \theta^2) \quad (3)$$

$$\begin{aligned} \dot{V} &= r\dot{r} + \theta\dot{\theta} \\ &= -rv \cos(\delta) + \theta v \sin(\delta) \end{aligned} \quad (4)$$

Fixing δ to enforce a virtual steering control as in [3],

$$\delta = \arctan(-k_1\theta). \quad (6)$$

The derivative of the Lyapunov function becomes:

$$\dot{V} = -rv \cos(\arctan(-k_1\theta)) + \theta v \sin(\arctan(-k_1\theta)). \quad (7)$$

Because $\theta \in (-\pi, \pi]$, $v > 0$, and $r \geq 0$, $\dot{V} < 0 \forall \theta$, $r \neq 0$; thus, the steering control asymptotically drives $r, \theta \rightarrow 0$. By choosing $v = k_3 r$ in some neighborhood of $r = 0$ for positive constant k_3 , the singularity at $r = 0$ is removed and the system is globally asymptotically stable. Furthermore,

the virtual steering control simultaneously drives the robot to the final position while aligning its heading with the positive global x -axis.

To drive the state to zero through the steering control, the heading must be driven such that $\delta = \arctan(-k_1\theta)$. An objective z_1 is defined:

$$z_1 \equiv \delta - \arctan(-k_1\theta). \quad (8)$$

The derivative is calculated to yield:

$$\dot{z}_1 = \left(1 + \frac{k_1}{1 + (k_1\theta)^2}\right) \frac{v}{r} \sin(z_1 + \arctan(-k_1\theta)) + \omega. \quad (9)$$

Feedback linearization can be used to achieve this objective by choosing the angular velocity

$$\omega = -\frac{v}{r} \left[k_2 z_1 + \left(1 + \frac{k_1}{1 + (k_1\theta)^2}\right) \sin(z_1 + \arctan(-k_1\theta)) \right], \quad (10)$$

the objective dynamics are globally exponentially stable with the following form:

$$\dot{z}_1 = -k_2 \frac{v}{r} z_1, \quad (11)$$

where k_1 and k_2 can be chosen to increase or decrease how aggressive the steering control is. This is modelled along with other constraints by defining:

$$V_2 = \frac{1}{2} z_1^2. \quad (12)$$

A second steering controller is designed to drive the state away from the origin and is defined by the following objective:

$$z_2 = \delta - \arctan(k_1\beta). \quad (13)$$

The derivative of z_2 is similarly computed as with z_1 , and is given by

$$\dot{z}_2 = \dot{\delta} \quad (14)$$

By choosing the angular velocity input w to drive the heading $\delta = \arctan(-k_1\theta)$, steering control, which drives the state to zero, is achieved. Similarly, choosing the angular velocity input w to drive the heading $\delta = \arctan(k_1\beta)$ causes the robot to drive away from zero. Thus, there exists a control input that satisfies the control Lyapunov function $\dot{V} < 0$; furthermore, following Definition 3 in [1], the control Lyapunov function is exponentially stabilizing if it satisfies

$$\inf_{u \in U} [\dot{V} + \gamma V] \leq 0 \quad (15)$$

for some constant $\gamma > 0$, so choosing the correct inputs u will achieve the desired exponential stabilization. According to [2], by picking a soft constraint relaxation variable λ , the condition for exponential stabilization shown in (15) is relaxed and can be enforced simultaneously along with a single hard constraint. Because control of the robotic system requires either $\dot{z}_1 = -k_2 \frac{v}{r} z_1$ or $\dot{z}_2 = -k_2 \frac{v}{r} z_2$ and $\dot{V} < 0$, two soft constrained expressions must be satisfied at a time to achieve exponential stabilization of the state to zero.

B. Barrier Functions

As in [2] and [1], control barrier functions are a logical way of enforcing safety constraints in navigation from a nonlinear control perspective.

A zeroing barrier function is used to drive the robot away from obstacles if it leaves the safe set (e.g. the robot gets within an unsafe threshold distance of the obstacle). The safe set C is described by the following expressions:

$$C = x \in \mathbb{R}^3 | z_b \geq z_{safe} \quad (16)$$

$$\partial C = x \in \mathbb{R}^3 | z_b = z_{safe} \quad (17)$$

$$\text{Int}(C) = x \in \mathbb{R}^3 | z_b > z_{safe} \quad (18)$$

As presented in [4], a barrier function is a zeroing barrier function $h(x)$ if $\dot{h}(x) \leq \gamma h(x)$. For the purpose of avoiding obstacles in the event that the robot leaves the safe set, a zeroing barrier function was chosen with the following form:

$$h(x) = \sqrt{(-v \cos(\theta) - b_x)^2 + (v \sin(\theta) - b_y)^2}. \quad (19)$$

The derivative of $h(x)$ is given by the following expression:

$$\dot{h}(x) = \frac{1}{h(x)} [(-v \cos(\delta)(r - b_y \sin(\theta) + b_x \cos(\theta)) - v \sin(\delta)(b_x \sin(\theta) + b_y \cos(\theta)))] \quad (20)$$

A barrier function was chosen to prevent the robot from leaving the safe set. According to Definition 2 in [2], the barrier function B is a control barrier function if

$$\inf_{u \in U} \left[\dot{B} + \frac{\gamma}{B} \right] \leq 0. \quad (21)$$

Thus, by satisfying the equation $\dot{B} \leq 1/B$, the barrier function will prevent the robot from leaving the safe set. The barrier function was chosen to be

$$B = \frac{1}{h(x) - z_{safe}}, \quad (22)$$

and its derivative is given by

$$\dot{B} = \frac{-\dot{h}(x)}{(h(x) - z_{safe})^2}. \quad (23)$$

C. QP Based Controller

Quadratic programming is used to find control inputs that simultaneously satisfy both the soft and hard constraints of the system. To minimize control inputs, the cost function $\mathbf{u}^T H \mathbf{u} + F^T \mathbf{u}$ is chosen. The QP is given by the following expression:

$$u^*(x, z) = \arg \min \frac{1}{2} \mathbf{u}^T H \mathbf{u} + F^T \mathbf{u} \quad (24)$$

$$\mathbf{u} = \begin{bmatrix} v \\ \omega \\ \lambda_1 \\ \lambda_2 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} \in \mathbb{R}^6$$

$$\text{s.t.} \quad \dot{V}_1 - \alpha_1 V_1 \leq \lambda_1 \quad (25)$$

$$\dot{V}_2 - \alpha_2 V_2 \leq \lambda_2 \quad (26)$$

$$\dot{h}(x) - \alpha h(x) \leq \lambda_1 \quad (27)$$

$$\dot{B} \leq \gamma/B \quad (28)$$

$$c_a \leq v \leq c_b \quad (29)$$

$$c_c \leq \omega \leq c_d \quad (30)$$

$$\alpha_1, \alpha_2 \leq 0 \quad (31)$$

In (24), the sparse matrix $H \in \mathbb{R}^{6 \times 6}$ has 1 in the first four elements of the diagonal, and the last two elements of the sparse vector $F^T \in \mathbb{R}^{1 \times 6}$ are 10 and 1. The last two elements in the diagonal of H were chosen to be zero to ensure that the negative quadratic cost of the slack variables α_1 and α_2 is as large as possible. Conversely, the last two elements of F^T were chosen to ensure that α_1 and α_2 were negative to minimize the cost.

The control Lyapunov functions V_1 and V_2 are the Lyapunov functions corresponding to the objectives of driving the state to zero and enforcing the steering control respectively; the equations are given by

$$V_1 = \frac{1}{2}(r^2 + \theta^2) \quad (32)$$

and

$$V_2 = \frac{1}{2}z_*^2. \quad (33)$$

z_* in V_2 represents the different steering objectives z_1 and z_2 in (8) and (13). Because there are two steering objectives, two QP control laws are required: one which drives the state towards the target while avoiding obstacles, and the other that drives the state in such a way that the robot steers away from obstacles. Toggling between steering objectives is dependent on the obstacle measurement. If the measurement of the distance between the robot and the obstacle was less than the minimum safe distance, then the steering objective z_2 is used. Otherwise, z_1 is used, and $\alpha_1 = 0$ is additionally enforced. $h(x)$ and B in the QP controller are given in equations (20) and (22).

III. RESULTS

MATLAB simulation was used to verify that the navigation control strategy using the QP controller enabled the robot to navigate to the desired position while avoiding obstacles. Figure 2 shows that the navigation control strategy is capable of driving the robot to the final position while avoiding obstacles from a variety of starting locations. Figure 3 shows that the navigation control strategy enables safe navigation between target waypoints generated through a probabilistic roadmap.

Implementing the navigation control law in the ROS framework, Gazebo simulation, shown in Figure 4, was used to corroborate the results of the MATLAB simulation. Following success in Gazebo, robot experiments were performed on the Segway robot “Jeeves”. The Hokuyo laser scans are subscribed to and the appropriate processing is performed. The robot

is commanded to drive to a set point and the optimal u is computed at every time step. The dynamics is propagated forward using the odometry. To add the obstacle, a human moved into the operation range of the obstacle as can be seen in Figure 5 and ensured that the robot did not break any safety constraints. A video showing one experimental run can be seen at <https://youtu.be/uhTiYyZxtfc>. The video shows that the robot maintains its safe distance from the human and tries to avoid the human, when the human walks out of the radius of influence the robot is free to move at greater speeds again.

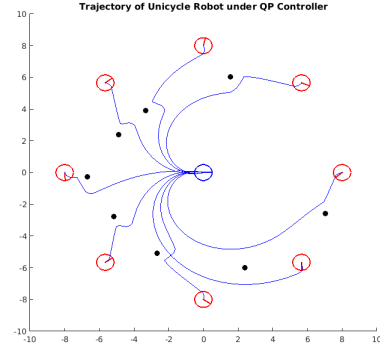


Fig. 2: The figure above shows the same initializations at eight different starting locations and the trajectories followed by the robot to the origin to show that the robot can generate a trajectory from an arbitrary initial condition to the origin while avoiding the obstacle.

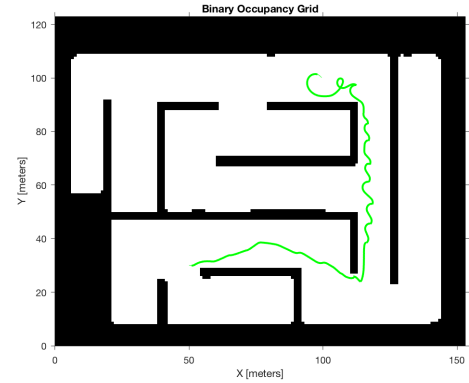


Fig. 3: Safe Navigation under QP Controller with Barrier Functions

IV. CONCLUSION

We show that the nonlinear controller is generalizable to multiple robots and is provably safe. The theoretical analysis shows that the controller stabilizes the system while obeying the “safety” rules. The experimental analysis shows that the system can be implemented on a real world system and the results are consistent with the claims. The developed system is almost self-contained and does not require many parameter

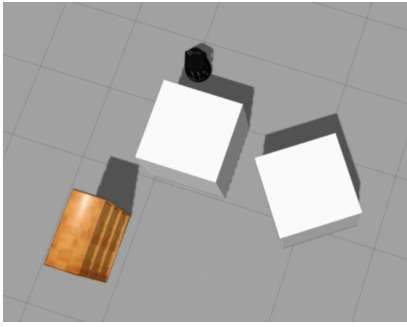


Fig. 4: Gazebo Simulation Navigation



Fig. 5: Real robot experiment with the segway RMP robot. The robot is stopped at a safe distance from the human obstacle.

adjustments. For future work, we propose more complex modelling of the obstacles to better capture the nature of the obstacle. For example, the safety barrier around a human is different from that around a wall. It is more “socially” acceptable to drive close to a wall but not as acceptable to be that close to a human.

REFERENCES

- [1] Aaron D. Ames, Kevin Galloway, Koushil Sreenath, and Jessy W. Grizzle. Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. In *IEEE Transactions on Automatic Control (Volume:59 , Issue: 4)*, pages 876–891. IEEE, 2014.
- [2] Aaron D. Ames, Jessy W. Grizzles, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conference on Decision and Control*, pages 6271–6278. IEEE, 2014.
- [3] Jong Jin Park and Benjamin Kuipers. A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4896–4902. IEEE, 2011.
- [4] Xiangru Xu, Paulo Tabuada, Jessy W. Grizzle, and Aaron D. Ames. Robustness of control barrier functions for safety critical control. In *Analysis and Design of Hybrid Systems(Volume 48)*, pages 54–61, 2015.